UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/734,849	12/13/2000	Kazuya Koyama	070639/0133	4741

22428 7590 12/31/2003

FOLEY AND LARDNER
SUITE 500
3000 K STREET NW
WASHINGTON, DC 20007

EXAMINER	
SHRADER, LAWRENCE J	
ART UNIT	PAPER NUMBER
2124	4

DATE MAILED: 12/31/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Office Action Summary	Application No. 09/734,849	Applicant(s) KOYAMA, KAZUYA
	Examiner Lawrence Shrader	Art Unit 2124

-- *The MAILING DATE of this communication appears on the cover sheet with the correspond nce address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 12 December 2000.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-34 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-34 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☒ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 4 .	6) ☐ Other: .

DETAILED ACTION

Specification

The abstract of the disclosure is objected to because the length exceeds 150 words. Correction is required. See MPEP § 608.01(b).

Applicant is reminded of the proper language and format for an abstract of the disclosure.

The abstract should be in narrative form and generally limited to a single paragraph on a separate sheet within the range of 50 to 150 words. It is important that the abstract not exceed 150 words in length since the space provided for the abstract on the computer tape used by the printer is limited. The form and legal phraseology often used in patent claims, such as "means" and "said," should be avoided. The abstract should describe the disclosure sufficiently to assist readers in deciding whether there is a need for consulting the full patent text for details.

The language should be clear and concise and should not repeat information given in the title. It should avoid using phrases which can be implied, such as, "The disclosure concerns," "The disclosure defined by this invention," "The disclosure describes," etc.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1 – 3, 5 – 7, 9 – 14, 16; 17 – 19, 21, 23, 24; 26 – 28, 30, 32, and 33 is rejected under 35 U.S.C. 102(e) as being anticipated by Davidson et al., U.S. Patent 6,042,614 (admitted prior art, hereinafter referred to as Davidson).

In regard to claim 1:

"A distributed debugger system, which debugs a distributed system which is configured by a program run on plural computers, comprising:

a program manager for executing a management from a predetermined computer to other computers via a network interconnecting said plural computers, said management being related to the setting status and execution status of a debug object program executed on each of said plural computers."

Davidson discloses a distributed debugger system, configured to run on a plurality of computers (Abstract) employing a program manager (column 7, line 65 to column 8, line 8) executing from a predetermined computer to other computers via a network (column 4, line 66 to column 5, line 1), wherein the management relates to setting status and execution status of the debug program (column 8, lines 11 – 36). These functions run on multiple clients with multiple servers (e.g., see Figure 7; column 8, lines 57 – 67; column 9, line 65 to column 10, line 12)

In regard to claim 2, incorporating the rejection of claim 1:

"...wherein said program manager comprises:

at least one of controllers for receiving instructions from a user;

Davidson discloses a debugger tool with a user interface (column 7, line 65 to column 8, line 11).

at least one of executors connected to said debug object program;

See Davidson column 8, lines 12 – 27.

each of said controllers including a setting-status manager for managing the setting of each of debuggers constructing said distributed debugger system and communication means for communicating with other controllers or said executors;

Davidson discloses a means to set status and a means to communicate with other controllers (column , line 65 to column 8, line 20; column 11, line52 to column 12, line 24).

each of said executors including a setting-status manager for managing the setting of each of debuggers constructing said distributed debugger system and communication means for communicating with other controllers or executors;

The Davidson sets breakpoints in the debuggers (column 8, lines 12 – 48; column 12, lines 16 – 20).

said setting-status manager changing its setting content when a change in setting is instructed and then notifying a setting-status manager in other controller or said executor of the changed content using said communication means, while said setting status manager changes its setting content in response to notification;

See Davidson column 12, lines 16 – 19.

whereby said distributed system to be debugged is debugged using the same setting on all computers in which said distributed debugger operates."

See Davidson column 12, lines 16 – 19.

In regard to claim 3, incorporating the rejection of claim 2:

"... wherein said program manager comprises:

at least one of controllers for receiving instructions from a user;

Davidson discloses a debugger tool with a user interface (column 7, line 65 to column 8, line 11).

at least one of executors connected to said debug object program;

See Davidson column 8, lines 12 – 27.

each of said controllers including an execution-status manager for managing an execution status of each of debuggers constructing said distributed debugger system, and communication means for communicating with other controllers or executors;

See Davidson column 8, lines 9 – 20 wherein the dbx engine allows communication to determine execution status.

each of said executors including said execution-status manager for managing the execution status of said debugger, a process manager for managing a debug object program, and communication means for communicating with other controllers or executors;

See Davidson column 8, lines 2 – 20.

wherein said execution-status manager changes its setting content when being instructed to change the execution status; said communication means notifies said execution-status manager in other controller or execution-status manager of the changed content; said execution-status manager changes its status in response to the notification and instructs said process manager to instruct a change of operation; whereby the same execution status is maintained on all computers on which said distributed debugger system operates."

In the Davidson invention, when the target program crashes the variables and set-points are read and notification sent to the dbx engines (column 8, lines 37, column 9, line 65 to column 10, line 12).

In regard to claim 5, incorporating the rejection of claim 2:

"...wherein said program manager and said debug object manager are realized on the same distributed system construction foundation and wherein program manager uses a communication function provided by said distributed system construction foundation."

Davidson teaches a communication function provided by the debugger and the underlying CORBA system (column 8, line 57 – column 9, line 4).

In regard to claim 6, incorporating the rejection of claim 2:

"...wherein said program manager comprises:

said setting-status manager for holding as a setting status a debug object program activation method;

Davidson teaches that the debugging system operates at the object code level (column 8, lines 15 – 25).

a remote debugger activator for activating said executor on a remote computer;

See Davidson column 10, lines 13 – 34.

a user interface for interpreting a debug object program from a user and a specification of an execution start place thereof;

See Davidson column 8, lines 9 – 20 for the user interface, and column 10, lines 13 – 20 for the start specification.

said user interface receiving a specification of said debug object program from said user and then notifying said setting-status manager of the specification;

See Davidson column 8, lines 25 for notification of status.

said user interface receiving the specification of said execution start place of said debug object program from said user and instructing said executor specified by said remote debugger activator to activate said executor;

The dbx engine reads information as needed (column 8, lines 12 – 27).

said remote debugger activator activating said executor on a specified computer;

See Davidson column 10, lines 1 – 6.

said user interface instructs said activated executor to start the execution of said debug object program;

See Davidson column 12, lines 16 – 19.

said process manager in said executor instructed acquiring information about a debug object program held in said setting-status manager to start execution of said debug object program;

See Davidson column 12, lines 16 – 19.

whereby execution of a debug object program is started on a computer different from the computer operated by said user."

See Davidson column 12, lines 16 – 19.

In regard to claim 7, incorporating the rejection of claim 1:

"...wherein said program manager comprises:

at least one of controllers for receiving instructions from a user;

See Davidson column 8, lines 9 – 13.

at least one of executors connected to said debug object program;

See Davidson columns, lines 16 – 27.

each of said controllers including an execution-status manager for managing an execution status of each of debuggers constructing said distributed debugger system, and communication means for communicating with other controllers or executors;

See Davidson column 10, lines 1 – 6, 13 – 20.

each of said executors including said execution-status manager for managing the execution status of said debugger, a process manager for managing a debug object program, and communication means for communicating with other controllers or executors;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

wherein said execution-status manager changes its setting content when being instructed to change the execution status; said communication means notifies said execution-status manager in other controller or execution-status manager of the changed content; and said execution-status manager changes its status in response to the notification and instructs said process manager to instruct a change of operation; whereby the same execution status is maintained on all computers on which said distributed debugger system operates."

See Davidson, which at least inherently performs all the claimed functions (column 8, lines 9 – 27, column 10, lines 1 – 6).

In regard to claim 9, incorporating the rejection of claim 8:

"...wherein said program manager and said debug object manager are realized on the same distributed system construction foundation and wherein program manager uses a communication function provided by said distributed system construction foundation."

Davidson teaches a communication function provided by the debugger and the underlying CORBA system (column 8, line 57 – column 9, line 4).

In regard to claim 10, incorporating the rejection of claim 7:

"...wherein said program manager interprets a request for changing the status of a debug object program, from a user, and instructs said execution-status manager to change the status."

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

In regard to claim 11, incorporating the rejection of claim 10:

"...wherein said program manager and said debug object manager are realized on the same distributed system construction foundation and wherein program manager uses a communication function provided by said distributed system construction foundation."

Davidson teaches a communication function provided by the debugger and the underlying CORBA system (column 8, line 57 – column 9, line 4).

In regard to claim 12, incorporating the rejection of claim 7:

"...wherein said process manager in said executor within said program manager changes its operation according to the operation of a debug object program and changes the status of said execution-status manager within the same executor based on the changed content."

See Davidson column 10, lines 13 – 23.

In regard to claim 13, incorporating the rejection of claim 12:

"...wherein said program manager and said debug object manager are realized on the same distributed system construction foundation and wherein program manager uses a communication function provided by said distributed system construction foundation."

Davidson teaches a communication function provided by the debugger and the underlying CORBA system (column 8, line 57 – column 9, line 4).

In regard to claim 14, incorporating the rejection of claim 7:

"...wherein said program manager and said debug object manager are realized on the same distributed system construction foundation and wherein program manager uses a communication function provided by said distributed system construction foundation."

Davidson teaches a communication function provided by the debugger and the underlying CORBA system (column 8, line 57 – column 9, line 4).

In regard to claim 16, incorporating the rejection of claim 1:

"...wherein said program manager and said debug object manager are realized on the same distributed system construction foundation and wherein program manager uses a communication function provided by said distributed system construction foundation."

Davidson teaches a communication function provided by the debugger and the underlying CORBA system (column 8, line 57 – column 9, line 4).

In regard to claim 17:

"A debugging method, suitable in use for a distributed debugger system that debugs a distributed system configured of a program which runs on plural computers, said method comprising the step of:

implementing a program management from a predetermined computer to other computers via a network interconnecting said computers, said program management being related to the setting status and execution status of a debug object program to be executed on each of said computers."

Davidson discloses a distributed debugger system, configured to run on a plurality of computers (Abstract) employing a program manager (column 7, line 65 to column 8, line 8) executing from a predetermined computer to other computers via a network (column 4, line 66 to column 5, line 1), wherein the management relates to setting status and execution status of the debug program (column 8, lines 11 – 36).

In regard to claim 18, incorporating the rejection of claim 17:

"...wherein said program management step comprises the sub-steps of:

transmitting an instruction of a setting change from a user or other computers to each of said debuggers constructing said distributed debugger system;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

changing the setting in response to said instruction;

The Davidson invention uses a Remote Breakpoint instruction (column 9, line 61), which is used to change a setting.

deciding whether or not said instruction has come from other computers;

See Davidson column 11, lines 8 – 17 to identify the instruction location.

notifying said debuggers on said other computers of said instruction via communications when said instruction has not come via communications;

Davidson discloses inter dbx communication (column 10, lines 1 – 6).

whereby a distributed debugger system to be debugged is debugged using the same settings on all computers on which said distributed debugger system operates."

See Davidson column 10, lines 13 – 30.

In regard to claim 19, incorporating the rejection of claim 18:

"...wherein each of debuggers constructing said distributed debugger system includes the steps of:

receiving an instruction of a change in execution status;

See Davidson column 8, lines 9 – 20.

deciding whether or not said instructed status corresponds to a change to the same status as a current status;

In any debugging system, at least the user implementing the GUI would inherently decide whether or not the status changes.

changing the status when the instructed status is not the same as the current status;

The Davidson invention determines if a program crashes and a new action is taken accordingly (column 8, lines 13 – 20).

deciding whether or not said execution status change is instructed according to a change of the management status of a debug object program;

See Davidson for determination of whether or not a breakpoint is reached (column 8, lines 38 – 48).

instructing said debugger on other computer to change the execution status via communications when the status change is instructed due to a change of the management status of said debug object program;

Davidson teaches communication of status among components of the distributed debugging system (column 9, line 54 to column 10, line 6).

deciding whether or not a debugger is said debugger which is managing said debug object program when the status change is not instructed due to a change of the management status of said debug object program;

See Davidson for determination of whether or not a breakpoint is reached (column 8, lines 38 – 48).

changing the management status of said debug object program in accordance with the status changed when a debugger is said debugger which is managing said debug object program;

See Davidson column 12, lines 24 regarding status changes.

whereby the same execution status is maintained on all computers on which said distributed debugger system operates."

In the Davidson invention, when the target program crashes the variables and set-points are read and notification sent to the dbx engines (column 8, lines 37, column 9, line 65 to column 10, line 12).

In regard to claim 21, incorporating the rejection of claim 17:

"...wherein each of debuggers constructing said distributed debugger system includes the steps of:

receiving an instruction of a change in execution status;

See Davidson column 8, lines 9 – 20.

deciding whether or not said instructed status corresponds to a change to the same status as a current status;

In any debugging system, at least the user implementing the GUI would inherently decide whether or not the status changes.

changing the status when the instructed status is not the same as the current status;

The Davidson invention determines if a program crashes and a new action is taken accordingly (column 8, lines 13 – 20).

deciding whether or not said execution status change is instructed according to a change of the management status of a debug object program;

See Davidson for determination of whether or not a breakpoint is reached (column 8, lines 38 – 48).

instructing said debugger on other computer to change the execution status via communications when the status change is instructed due to a change of the management status of said debug object program;

Davidson teaches communication of status among components of the distributed debugging system (column 9, line 54 to column 10, line 6).

deciding whether or not a debugger is said debugger which is managing said debug object program when the status change is not instructed due to a change of the management status of said debug object program;

See Davidson for determination of whether or not a breakpoint is reached (column 8, lines 38 – 48).

changing the management status of said debug object program in accordance with the status changed when a debugger is said debugger which is managing said debug object program;

See Davidson column 12, lines 24 regarding status changes.

whereby the same execution status is maintained on all computers on which said distributed debugger system operates."

In the Davidson invention, when the target program crashes the variables and set-points are read and notification sent to the dbx engines (column 8, lines 37, column 9, line 65 to column 10, line 12).

In regard to claim 23, incorporating the rejection of claim 21:

"...wherein each of debuggers constructing said distributed debugger system, comprises the steps of:

receiving an instruction of a status change from a user;

Davidson discloses the user interfacing with a dbx engine (column 8, lines 9 – 16 providing the user with an interactive debugger.

interpreting said instruction and then changing the execution status."

The interactive debugger of Davidson inherently interprets the user's instructions to change execution status.

In regard to claim 24, incorporating the rejection of claim 21:

"...wherein each of debuggers constructing said distributed debugger system, comprises the steps of:

monitoring an operation status of a debug object program;

The interactive feature of the Davidson invention provides the user a means to monitor the status of the program (column 8, lines 9 – 16).

changing the management status of said debug object program when specific requirements are satisfied during monitoring;

The debugger monitors the program and the state changes if a break point is encountered, e.g., single-step operation (column 8, lines 25 – 36).

instructing an execution status change in accordance with a management status change."

The execution changes to a step mode with the change of the debugger status (column 8, lines 25 – 36).

In regard to claim 26:

"A recording medium, on which a control program for a distributed debugger system for debugging a distributed system constructed by a program which runs on plural computers is recorded, said recording medium recording a program management step of executing a management from a specific computer to other computers via a network interconnecting plural computers, said management being related to the setting status and execution status of a debug object program executed on each of said computers."

Claim 26 is rejected for the same reason put forth in the rejection of claim 17 (a corresponding system).

In regard to claim 27 (a recording medium), incorporating the rejection of claim 26, is rejected for the same corresponding reasons put forth in the rejection of claim 18 (the debugging method).

In regard to claim 28 (a recording medium), incorporating the rejection of claim 27, is rejected for the same corresponding reasons put forth in the rejection of claim 19 (the debugging method).

In regard to claim 30 (a recording medium), incorporating the rejection of claim 26, is rejected for the same corresponding reasons put forth in the rejection of claim 21 (the debugging method).

In regard to claim 32 (a recording medium), incorporating the rejection of claim 30, is rejected for the same corresponding reasons put forth in the rejection of claim 23 (the debugging method).

In regard to claim 33 (a recording medium), incorporating the rejection of claim 30, is rejected for the same corresponding reasons put forth in the rejection of claim 24 (the debugging method).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 4, 8, 15, 20, 22, 25, 29, 31, and 34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Davidson et al., U.S. Patent 6,042,614, as applied to claims 1 and 17 above, in view of the SPARCworks "Sun Product Documentation."

In regard to claim 4, incorporating the rejection of claim 2:

"...wherein said program manager comprises:

at least one of controllers for receiving instructions from a user;

Davidson discloses a debugger tool with a user interface (column 7, line 65 to column 8, line 11).

at least one of executors connected to said debug object program;

See Davidson column 8, lines 12 – 27.

each of said controllers including a user interface for interpreting a request for displaying the status of a debug object program from a user and displaying the results; place decision means for specifying one or more existence places in the status specified based on the nature of the status when the status of a debug object program is specified and based on the execution status of a debug object program acquired from said process manager, and communication means for communicating with other controller or executor;

See the GUI interface in Davidson e.g., column 8, lines 9 – 20, Figure 9, also the ability to communicate with all dbx engines in the system (column 10 lines 1 – 6).

each of executors including a process manager for managing a debug object program and communication means for communicating with other controller or executor;

See Davidson column 10, lines 1 – 6 for communication means for the debugger tool to communicate with various components.

said user interface inquiring said place decision means in response to a request for displaying the status of a debug object program from a user and then acquiring one or more existence places of the status and transmitting a status capture request to said process managers at all existence places via said communication means;

The user interfacing with the GUI is able to determine statuses and locations of the status (column 10, lines 1 – 6).

said process manager checking the execution status of a debug object program under management in response to a status capture request and transmitting results to said user interface at the request source; said user interface receiving one or more results and then outputting said results in a batch mode to said user, whereby said user acquires the status of a distributed system to be objected, without recognizing its existence place."

Davidson teaches checking execution status and transmitting the results to the user interface (column 8, lines 9 – 31; column 10, lines 1 – 6), but does not teach outputting results to a batch file. However, the SPARCworks Sun Product Documentation , page 3 discloses collection of data commands in batch jobs. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to use the batch collection function of the modified SPARCworkstation used in the Davidson invention as described in the SPARCworks documentation so that the user has the ability to analyze data offline.

In regard to claim 8, incorporating the rejection of claim 7:

"...wherein said program manager comprises:

at least one of controllers for receiving instructions from a user;

See Davidson column 8, lines 9 – 13.

at least one of executors connected to said debug object program;

See Davidson columns, lines 16 – 27.

each of said controllers including a user interface for interpreting a request for displaying the status of a debug object program from a user and displaying the results; place decision means for specifying one or more existence places in the status specified based on the nature of the status when the status of a debug object program is specified and based on the execution status of a debug object program acquired from said process manager, and communication means for communicating with other controller or executor;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

each of executors including a process manager for managing a debug object program and communication means for communicating with other controller or executor;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

said user interface inquiring said place decision means in response to a request for displaying the status of a debug object program from a user and then acquiring one or more existence places of the status and transmitting a status capture request to said process managers at all existence places via said communication means;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6, and 13 – 20.

said process manager checking the execution status of a debug object program under management in response to a status capture request and transmitting results to said user interface at the request source;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

said user interface receiving one or more results and then outputting said results in a batch mode to said user, whereby said user acquires the status of a distributed system to be objected, without recognizing its existence place."

Davidson teaches checking execution status and transmitting the results to the user interface (column 8, lines 9 – 31; column 10, lines 1 – 6), but does not teach outputting results to a batch file. However, the SPARCworks Sun Product Documentation , page 3 discloses collection of data commands in batch jobs. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to use the batch collection function of the modified SPARCworkstation used in the Davidson invention as described in the SPARCworks documentation so that the user has the ability to analyze data offline.

In regard to claim 15, incorporating the rejection of claim 1:

"...wherein said program manager comprises:

at least one of controllers for receiving instructions from a user;

See Davidson column 8, lines 9 – 13.

at least one of executors connected to said debug object program;

See Davidson columns, lines 16 – 27.

each of said controllers including a user interface for interpreting a request for displaying the status of a debug object program from a user and displaying the results; place decision means for specifying one or more existence places in the status specified based on the nature of the status when the status of a debug object program is specified and based on the execution status of a debug object program acquired from said process manager, and communication means for communicating with other controller or executor;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

each of executors including a process manager for managing a debug object program and communication means for communicating with other controller or executor;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

said user interface inquiring said place decision means in response to a request for displaying the status of a debug object program from a user and then acquiring one or more existence places of the status and transmitting a status capture request to said process managers at all existence places via said communication means;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

said process manager checking the execution status of a debug object program under management in response to a status capture request and transmitting results to said user interface at the request source;

See Davidson column 8, lines 1 – 20; column 10, lines 1 – 6.

said user interface receiving one or more results and then outputting said results in a batch mode to said user, whereby said user acquires the status of a distributed system to be objected, without recognizing its existence place."

Davidson teaches checking execution status and transmitting the results to the user interface (column 8, lines 9 – 31; column 10, lines 1 – 6), but does not teach outputting results to a batch file. However, the SPARCworks Sun Product Documentation , page 3 discloses collection of data commands in batch jobs. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to use the batch collection function of the modified SPARCworkstation used in the Davidson invention as described in the SPARCworks documentation so that the user has the ability to analyze data offline.

In regard to claim 20, incorporating the rejection of claim 18:

"...wherein each of said debuggers constructing said distributed debugger system implements the steps of:

receiving an instruction for displaying a status from a user;

Davidson implements a GUI to display debugging information to the user (column 8, lines 9 – 15).

interpreting said instruction;

See Davidson column 8, lines 14 – 16 for interpreting the displayed information.

specifying one or more specified status existence places;

See Davidson column 8, lines 13 – 16 for determining the where a program crashes.

transmitting a status capture request to debuggers at all existence places;

See Davidson column 8, lines 13 – 16 and column 10, lines 1 - 6

acquiring the status of a debug object program by means of said debugger which has received said status capture request;

See Davidson column 8, lines 9 – 20, column 10, lines 1 – 6 for acquiring the program status by means of the debugger.

acknowledging the acquired status to said debugger being a request source;

See Davidson column 8, lines 16 – 20 and column 10, lines 1 – 6.

receiving all replies by means of said debugger being a request source;

See Davidson column 8, lines 9 – 20 and column 10, lines 1 – 6.

outputting contents of received replies in a batch mode to said user;

Davidson teaches checking execution status and transmitting the results to the user interface (column 8, lines 9 – 31; column 10, lines 1 – 6), but does not teach outputting results to a batch file. However, the SPARCworks Sun Product Documentation , page 3 discloses collection of data commands in batch jobs. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to use the batch collection function of the modified SPARCworkstation used in the Davidson invention as described in the SPARCworks documentation so that the user has the ability to analyze data offline.

whereby said user acquires the status inside said distributed system to be debugged, without recognizing the existence place."

See Davidson for seamless implementation regardless of where a function resides (column 10, lines 13 – 18).

In regard to claim 22, incorporating the rejection of claim 21:

"...wherein each of said debuggers constructing said distributed debugger system implements the steps of:

receiving an instruction for displaying a status from a user;

Davidson implements a GUI to display debugging information to the user (column 8, lines 9 – 15).

interpreting said instruction;

See Davidson column 8, lines 14 – 16 for interpreting the displayed information.

specifying one or more specified status existence places;

See Davidson column 8, lines 13 – 16 for determining the where a program crashes.

transmitting a status capture request to debuggers at all existence places;

See Davidson column 8, lines 13 – 16 and column 10, lines 1 - 6

acquiring the status of a debug object program by means of said debugger which has received said status capture request;

See Davidson column 8, lines 9 – 20, column 10, lines 1 – 6 for acquiring the program status by means of the debugger.

acknowledging the acquired status to said debugger being a request source;

See Davidson column 8, lines 16 – 20 and column 10, lines 1 – 6.

receiving all replies by means of said debugger being a request source;

See Davidson column 8, lines 9 – 20 and column 10, lines 1 – 6.

outputting contents of received replies in a batch mode to said user;

Davidson teaches checking execution status and transmitting the results to the user interface (column 8, lines 9 – 31; column 10, lines 1 – 6), but does not teach outputting results to a batch file. However, the SPARCworks Sun Product Documentation , page 3 discloses collection of data commands in batch jobs. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to use the batch collection function of the modified SPARCworkstation used in the Davidson invention as described in the SPARCworks documentation so that the user has the ability to analyze data offline.

whereby said user acquires the status inside said distributed system to be debugged, without recognizing the existence place."

See Davidson for seamless implementation regardless of where a function resides (column 10, lines 13 – 18).

In regard to claim 25, incorporating the rejection of claim 17:

"...wherein each of said debuggers constructing said distributed debugger system implements the steps of:

receiving an instruction for displaying a status from a user;

Davidson discloses the user interfacing with a dbx engine (column 8, lines 9 – 16 providing the user with an interactive debugger.

interpreting said instruction;

The interactive debugger of Davidson inherently interprets the user's instructions to change execution status.

specifying one or more specified status existence places;

A set break point specifies an existence place of a status (column 8, lines 13 – 16)

transmitting a status capture request to debuggers at all existence places;

The user interfacing with the GUI is able to determine statuses and locations of the status and communicate the information to al debuggers (column 10, lines 1 – 6).

acquiring the status of a debug object program by means of said debugger which has received said status capture request;

When a set point is captured, the status of the debug program is acquired

acknowledging the acquired status to said debugger being a request source;

See Davidson column 8, lines 16 – 20 and column 10, lines 1 – 6.

receiving all replies by means of said debugger being a request source;

See Davidson column 8, lines 9 – 20 and column 10, lines 1 – 6.

outputting contents of received replies in a batch mode to said user;

Davidson teaches checking execution status and transmitting the results to the user interface (column 8, lines 9 – 31; column 10, lines 1 – 6), but does not teach outputting results to a batch file. However, the SPARCworks Sun Product

Documentation , page 3 discloses collection of data commands in batch jobs. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to use the batch collection function of the modified SPARCworkstation used in the Davidson invention as described in the SPARCworks documentation so that the user has the ability to analyze data offline.

whereby said user acquires the status inside said distributed system to be debugged, without recognizing the existence place."

See Davidson for seamless implementation regardless of where a function resides (column 10, lines 13 – 18).

In regard to claim 29 (a recording medium), incorporating the rejection of claim 27, is rejected for the same corresponding reasons put forth in the rejection of claim 20 (the debugging method).

In regard to claim 31 (a recording medium), incorporating the rejection of claim 30, is rejected for the same corresponding reasons put forth in the rejection of claim 22 (the debugging method).

In regard to claim 34 (a recording medium), incorporating the rejection of claim 26, is rejected for the same corresponding reasons put forth in the rejection of claim 25 (the debugging method).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lawrence Shrader whose telephone number is (703) 305-8046. The examiner can normally be reached on M-F 08:00-16:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703) 305-9662. The fax phone number for the organization where this application or proceeding is assigned is (703) 746-7239.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Lawrence Shrader
Examiner
Art Unit 2124

December 18, 2003

[illegible]
KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100